Learning Online Discussion Structures by Conditional Random Fields

Hongning Wang, Chi Wang, ChengXiang Zhai, Jiawei Han
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana IL, 61801 USA
{wang296, chiwang1, czhai, hanj}@cs.uiuc.edu

ABSTRACT

Online forum discussions are emerging as valuable information repository, where knowledge is accumulated by the interaction among users, leading to multiple threads with structures. Such replying structure in each thread conveys important information about the discussion content. Unfortunately, not all the online forum sites would explicitly record such replying relationship, making it hard for both users and computers to digest the information buried in a discussion thread.

In this paper, we propose a probabilistic model in the Conditional Random Fields framework to predict the replying structure for a threaded online discussion. Different from previous replying relation reconstruction methods, most of which fail to consider dependency between the posts, we cast the problem as a supervised structure learning problem to incorporate the features capturing the structural dependency and learn their relationship. Experiment results on three different online forums show that the proposed method can well capture the replying structures in online discussion threads, and multiple tasks such as forum search and question answering can benefit from the reconstructed replying structures.

Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models - Statistical

General Terms

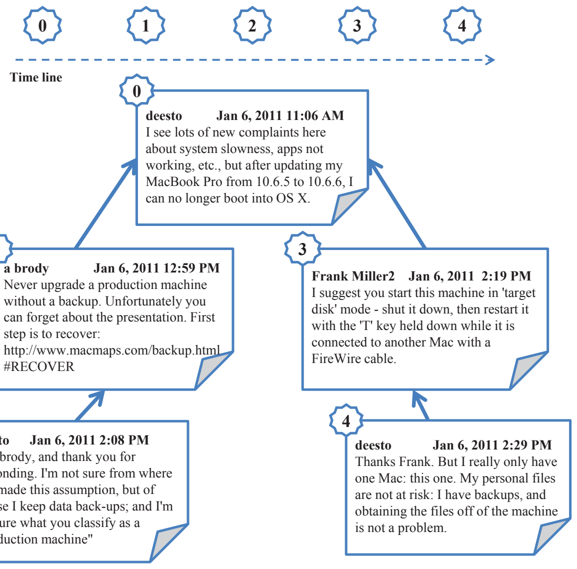
Algorithms, Measurement, Experimentation

Keywords

Threaded Discussion, Replying Relation Reconstruction, Structure Learning

1. INTRODUCTION

As the development of Web 2.0, more and more people take the advantage of online forum discussions to freely share and exchange their mind and knowledge. Valuable knowledge and information on various topics, e.g., sports, health, entertainment and etc., have been accumulated by this collaborative content contribution. More importantly, such knowledge can hardly be found in general web sites and encyclopedia, making forums a unique and valuable resource for extracting useful knowledge to facilitate other information seeking tasks, including forum search [2, 13], question answering [4, 6] and expert finding [21, 8].



deesto **Jan 6, 2011 11:06 AM**
I see lots of new complaints here about system slowness, apps not working, etc., but after updating my MacBook Pro from 10.6.5 to 10.6.6, I can no longer boot into OS X.

a brody **Jan 6, 2011 12:59 PM**
Never upgrade a production machine without a backup. Unfortunately you can forget about the presentation. First step is to recover: http://www.macmaps.com/backup.html #RECOVER

deesto **Jan 6, 2011 2:08 PM**
Hi a brody, and thank you for responding. I'm not sure from where you made this assumption, but of course I keep data back-ups; and I'm not sure what you classify as a "production machine"

Frank Miller2 **Jan 6, 2011 2:19 PM**
I suggest you start this machine in 'target disk' mode - shut it down, then restart it with the 'T' key held down while it is connected to another Mac with a FireWire cable.

deesto **Jan 6, 2011 2:29 PM**
Thanks Frank. But I really only have one Mac: this one. My personal files are not at risk: I have backups, and obtaining the files off of the machine is not a problem.

Figure 1: A sample threaded discussion from Apple Discussions

A typical online forum discussion originates from a root post, which initializes a topic for the following discussions, e.g., system failure in Mac OS X in Apple discussions as shown in Figure 1. The followers read existing messages and reply to the post they are informed of or most interested in. From temporal perspective, those replying posts form a chain structure, or thread (as shown in the time line in Figure 1). Replying posts can reply to any preceding post, forming branches of discussion as more users are joining in and making comments. As a result, the discussion thread grows and forms a tree structure from semantic perspective: one post has only one "reply-to" post, while one post can be replied to by multiple posts.

The semantic tree structure is helpful for both human to digest the discussion content and automatic method to extract useful information from it. From user's perspective, it would save a lot of effort to track and get involved in the discussion if the site provides a tree view of the current discussion structure. [13, 5] demonstrate that incorporating the replying relationship boosts the forum retrieval performance; Zhang et al. [21] show that ranking users by their replying relationship would further help to identify the experts in online communities than solely by their post counts.

Nevertheless, such tree structure is not always present in most of online discussion sites. Because phpBB (`http://www.phpbb.com`) and vBulletin (`http://www.vbulletin.com`), which are two most popular softwares to build the online discussion site, do not provide a threaded view as default [13], flat-view online community pages are much more prevalent. In the flat-view systems, the replying relation is not explicitly recorded (some of them contain partial content from the replied post as quotation), and all the posts in one thread are positioned by the chronological order.

In this circumstance, reconstructing the replying relationship from the flat-view discussion forums would be useful. However, this reconstruction task is quite challenging: 1) posts in a discussion thread are temporally dependent upon each other - users always read some of the previous messages before posting; 2) most of the posts are short and highly context dependent, solely examining the post content is inadequate to understand the replying relationship between the posts; 3) rich features are needed to perform better prediction - user's replying habits, friendship and even their online periods are all potential features helpful for analyzing the threaded discussions.

Several approaches have been proposed to address this kind of reconstruction task. A topic modeling based approach is proposed in [11], where the temporal relationship is incorporated into semantic modeling of threaded discussions. In [13], the reconstruction task is converted into a retrieval problem, in which the authors proposed various features to train a Ranking SVM model. Wang et al. [19] utilize a variant of Latent Semantic Analysis (LSA) method to overcome the sparseness in online discussion contents and identify the replying post pairs.

Previous work mainly focused on applying content analysis techniques to identify the replying relation between the posts. However, they do not take other factors, e.g. users interactions and structural dependency, into consideration. In this paper, we cast this replying relationship reconstruction problem as a structural learning task and propose a threadCRF model to solve it in the probabilistic model framework. Various features are incorporated into the proposed threadCRF to capture both long-range and short-range dependencies among the posts. To estimate the relative importance of those features, a supervised learning framework is employed. A set of novel evaluation metrics is introduced to access the quality of structural prediction. Experiment results on three different forum collections, Apple Discussion (`http://discussions.apple.com/`), Google Earth Community (`http://bbs.keyhole.com/`) and CNET forums (`http://forums.cnet.com/`), confirm the effectiveness of the proposed method: structure reconstruction quality gets encouraging improvement over the baseline methods; tasks like forum retrieval and community question answering benefit from the reconstructed replying structure. More importantly, the proposed method is adaptive and generalizes well when trained on different domains.

2. RELATED WORK

Much work has been done recently on extracting information from online forums. Ding et al. [4] extract contexts and answers for the questions from online forums; Shi et al. [15] analyze the user grouping behaviors in online forums; Cong et al. [3] detect the question-answer pairs in the threaded discussions.

Different from the traditional document repository, online forum discussions have their unique internal replying structures, which are crucial for correctly understanding the discussion contents. Xu et al. [20] illustrate as much as 75% of the links in a typical forum page points to noise pages like user profiles, login pages etc., and hence link-based algorithms like PageRank and HITS cannot be used effectively. Besides, they also demonstrate content-induced links are more helpful for ranking forum pages. Seo et al. [13] state that using thread structures to estimate language models improves forum search performance. In [8], Jurczyk et al. show that the linkage pattern can be used to help identify the quality of answers in the online community environment.

However, the replying structure of a threaded discussion is not always available, and some work has been proposed to reconstruct the conversional structures. Shen et al. [14] treat the reconstruction task as a content-based clustering problem and propose to use the clusters of posts as the subthreads in a threaded discussion. However, this method does not identify the explicit parent-child relations within each cluster. Wang et al. [18] use a graph-based connectivity matrix, in which both content similarity and temporal information are utilized, to recover thread structure. But the parameters are manually tuned in the proposed method. In [11], Lin et al. incorporate the temporal relationship into semantic modeling of threaded discussions based on the statistical topic model. Unfortunately, their model cannot directly predict the replying relationship, and they depend on an additional distance measure in the projected topic space to find the "closest" replying post pairs. In [13], Seo et al. converted this structure prediction task into a ranking problem: each post is considered as a query, and parent candidate posts are considered as the documents to be retrieved. Multiple features are utilized to learn a Ranking SVM model. But because Ranking SVM can only take the post's pairwise interaction into consideration, it fails to directly model the whole discussion structure.

In this paper, various kind of features, from content similarity and posting time gap to user interactions, are introduced in the proposed probabilistic model to capture both the long-range and short-range dependencies within a discussion thread. In addition, to judge the quality of the structure prediction, we design a set of new evaluation metrics, in which we emphasize more about the structure preservability in the good predicted result.

3. PROBLEM DEFINITION

Before introducing the proposed method in detail, we would first define some concepts that would be referred to in later discussions.

Formally, let $T = \{X_0, X_1, \ldots, X_N\}$ be a set of thread discussions from a particular online forum site; each thread X_n consists of m individual posts $\{p_0, p_1, \ldots, p_{m-1}\}$ arranged in the chronological order. In each post p_i of thread X_n, there are 5 attributes: 1) post ID, ranging from [0, m); 2) post content c_i, modeled as a bag of words; 3) author name u_i, the displayed author name for the post; 4) author ID a_i, a unique user identifier in the whole forum; and 5) posting time t_i, the system time stamp when the post is created. In this work, we only assume the above five general post attributes, which should be found in most of the online discussion forum sites; other site-specific attributes, such as post quotations and tree views, are not considered.

Definition (Root Post) Root post is the first post in one discussion thread according to its posting time. It initiates the topic of discussion in this thread.

Definition (Previous Post) Previous post of p_i is the post posted closest in prior to p_i in the chronological order.

Definition (Parent Post) Post p_i is said to be the parent post of p_j if and only if p_j is posted later than p_i and contains an immediate follow-up discussion of p_i. In the threaded-view forum site, this information is indicated by "in response to" or a reply tree. Such pairwise relationship is called "replying to" relation, which is asymmetric.

Definition (Thread Structure) The thread structure defined by the "replying to" relation between posts is strictly a tree: posts in X_n, except for the root post, can only and must have one parent post in X_n. Each path in this tree forms a self-consistent sub-threaded discussion. Such a structure is intuitively demonstrated in Figure 1.

Based on the above definitions, the task of reply relation reconstruction is equivalent to assigning every post p_i in thread X_n a parent post p_{y_i}, which makes every sub-threaded discussion consistent and coherent. We call such parent labeling sequence as Y_n for each X_n.

4. METHOD

Although the "replying to" relation between two posts is defined according to their semantic relations in Section 3, the interaction patterns among the replying posts are far beyond what can be observed merely from the post content. Because most of the posts are very short, content similarity is inadequate to infer the correct replying to relationship. Besides, the involvement of user interactions makes it even harder to predict the replying relation. For example, users are more likely to reply to the post which has replied to him/her before; root post tends to receive more replies than other posts. Another important character of threaded discussion is that the post contents are highly context dependent: the topic focus is evolving during the discussion, so that in order to understand the intension of a certain post it is often necessary to find the conversational information hidden in the related posts. Nevertheless, due to the intra-dependencies among the posts in one thread, such correspondence is not known until we recover the whole replying structure. Thus the difficulty arises from the intuition that every local prediction depends on all the other posts' reply-to assignment in the same thread. On the other hand, the internal dependencies within the replying structure provide regularization over each local prediction, which helps the overall prediction in turn: when we have high confidence on recovering some parts of the thread, the prediction on the less certain parts will become easier with the assumption that the local predictions can be propagated. Intuitively, the assumption holds if we can properly identify the dependency among the predictions. Besides, as there are various types of factors interacting and suggesting different level of dependency, including users, post contents and time, it is hard to determine their relative importance without learning from the data. Therefore, we need an appropriate mechanism to encode the structural dependency among such factors in a systematic way. To achieve this goal, we cast the thread replying relation reconstruction task as a structure learning problem, and propose a thread Conditional Random Field (threadCRF) model to solve it.

4.1 Thread Conditional Random Field

Our main focus in this paper is to exploit the dependencies among the posts, which can facilitate us to better identify the correct structures from the threaded discussion. Probabilistic graphical models provide a systematic methodology to describe and manipulate the short-range and long-range dependencies in both observed data and unknown predictions. By properly defining the features (or potential functions) in the joint/conditional probability, arbitrary dependencies could be encoded and their relative importance can be captured by the feature's weight.

In our reconstruction task, we are more interested in modeling the conditional probability of the replying relationship Y_n given the posts $\{p_0, p_1, \dots, p_{m-1}\}$ in thread X_n, i.e., $p(Y_n|X_n)$, than their joint probability $p(X_n, Y_n)$. Therefore, Conditional Random Fields (CRFs) [10], which is a family of probabilistic models for sequence segmentation and labeling problems, is the nature choice for solving this problem. By directly modeling the conditional probability, we can flexibly introduce any meaningful features, particularly dependent features of the observed sequence without having to model the distribution of those dependencies.

Following the basic concepts in traditional CRFs, we propose a threadCRF model to capture the dependency among the posts within one thread based on various kinds of features describing the interactions in both posts and authors, and estimate the weights for the designed features in a supervised manner.

In threadCRF, we define the conditional distribution over replying relationship given post sequence X_n as:

$$p(Y_n|X_n) \propto \exp\Big(\sum_{k=1}^{K} \lambda_k f_k(Y_n, X_n)\Big) \tag{1}$$

where $\{f_k(Y_n, X_n)\}_{k=1}^K$ is a set of features defined on the given thread X_n and its parent labeling sequence Y_n; $\{\lambda_k\}_{k=1}^K$ are the weights for the corresponding features.

Thus, given a model $\Theta = \{\lambda_k\}_{k=1}^K$, the replying relation reconstruction task in our threadCRF could be formulated as a Maximum a Posteriori (MAP) inference problem: for each given thread X, we aim to find the optimal replying structure Y', such that,

$$Y' = argmax_{Y \in \mathscr{Y}} p(Y|X, \Theta) \tag{2}$$

where $\mathscr{Y}$ is the set of all the possible replying structures for the given thread X.

Within such a framework, the key challenge is to define a proper set of features by which the dependencies among the posts could be explicitly captured. In this paper, we design

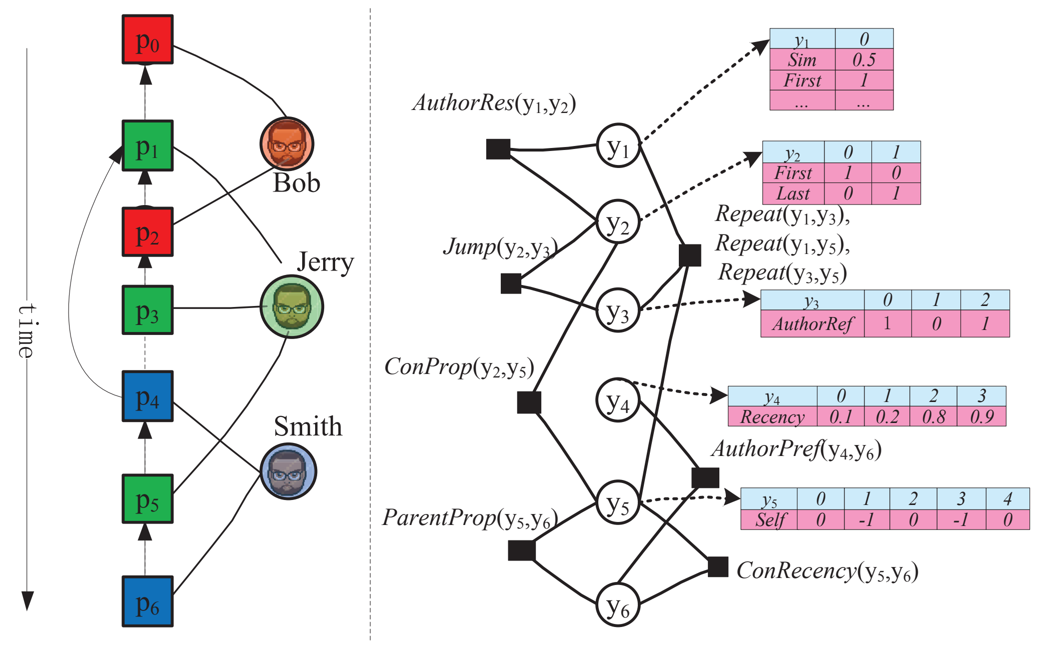


Figure 2: The graphical representation of the proposed threadCRF model, and an illustration of two kinds of features employed. On the left is an example of a threaded discussion; its corresponding factor graph induced by the proposed features is shown on the right. For each node in the factor graph, only a part of related features is shown due to space limit. Node features are listed in the table connected to the related variables by dotted line; edge features are represented by solid rectangles.

features to model both posts' replying patterns and users' interactions in the threaded discussion environment.

4.2 Features

We define two kinds of features, each of which can be formalized by some particular feature functions. The first kind of features depicts the local potential of replying relations, i.e., how likely post p_i replies to p_j if they are close in posting time. Such features can be defined by a function on p_i and its *own* parent assignment p_{y_i}, and outputs a certain confidence measure of this replying pair based on the observed attributes from p_i and p_{y_i}. The second kind of features captures the long-range dependency among the predictions, e.g., users prefer to reply to the post which has replied to him/her before. Such kind of feature functions for posts p_i and p_j not only depends on the observed attributes, but also on the unknown parent assignments for *both* p_i and p_j, since the parent prediction on one of these two posts would affect the parent assignment for the other. Using the language of probabilistic graphical models [9], we treat each post parent assignment as a random variable, and depict the dependency among the features by a factor graph, where the edges are connected by the defined features (as shown in Figure 2). In a factor graph, any complicated dependency can be decomposed into factor functions defined on edges. We can then propagate local predictions along the edges within the graph to find the global optimal prediction.

Previous methods [13] mostly operate on node features, and therefore they are performing isolated predictions for each post. The edge features proposed in this paper are new and can capture more complex dependency than those handled by the node features. Altogether, we propose six node features and seven edge features for our threadCRF. The proposed threadCRF incorporates both kind of features and learns the weights from the given training corpus.

4.2.1 Node features

Node features *only* depend on the observed attributes in post p_i and p_j, i.e., solely from the observed attributes of p_i and p_j, to determine how likely p_i is replying to p_j.

Content Similarity *Sim*(y_i): the content similarity between post p_i and p_{y_i}. One post tends to reply to another if they are talking about similar topics. We use the standard *TF-IDF* weighted cosine similarity [12], where *IDF* is calculated based on the number of posts containing the word in the given thread.

First Post *First*(y_i): a binary feature to test if post p_i is replying to the root post. In the threaded discussions, root post initiates the topic for the followers to discuss; as a result, the following posts tend to reply to the root post.

Last Post *Last*(y_i): a binary feature to test whether post p_i is replying to its previous post. Since in the conversational discussions, authors exchange their ideas in turn. One post in the dialog tends to directly reply to its previous post.

Author Reference *AuthorRef*(y_i): a binary feature to test whether post p_{y_i}'s author is mentioned in the content of post p_i. In the threaded discussions, especially in the flat-view forum site, to clearly point to the receiver, some posts would directly mention the name of the author of the post he/she is now replying to.

Time Recency *Recency*(y_i): the time proximity between posting time of post p_i and p_{y_i}. A small time difference between two posts could be an evidence of the replying relationship. We use the time difference between post p_i and the root post to normalize the time difference between the root post and p_{y_i}. We should note the feature ***Recency***(y_i) is different from ***Last***(y_i), since ***Recency***(y_i) is more sensitive to the actual time gap. When ***Recency***(y_i) approaches 0, it indicates there is a long period of time when no one has followed the previous discussion; as a result, the chance that p_i still replies to p_j would get lower though p_j is p_i's previous post.

Reply to Oneself $\textbf{\textit{Self}}(y_i)$: a binary feature to test if post p_i and p_{y_i} are posted by the same author. Since the discussion forum is a place to share and exchange opinions, we assume one author should seldom reply to his/her own post. We should admit that there exist such situations, where the authors want to clarify or expand his/her previous claim. But we believe those cases happen with an arguably small probability and thus assign negative credit for this feature.

4.2.2 Edge features

Edge features are defined over *two* parent assignments y_i and y_j (i>j) together with the observed attributes in p_i and p_j, i.e., (X, Y), to emphasize the dependencies between y_i and y_j. We can either boost or penalize their interactions to reflect our assumptions on the replying structure.

Repeat Reply $\textbf{\textit{Repeat}}(y_i, y_j)$: A binary feature to penalize the replying pattern where p_i and p_j reply to the same post, when p_i and p_j are written by the same author. This feature is designed to capture the intuition that users seldom reply to their own posts.

Jumping Reply $\textbf{\textit{Jump}}(y_i, y_j)$: A binary feature to penalize the replying pattern where p_i replies to an earlier post p_k by p_j's author a_j rather than its closest preceding post p_j. The intuition behind this feature is that in a conversational discussion, once people get replies from a recent post, he/she is unlikely to jump to an earlier post by the same author to start another conversation.

Author Response $\textbf{\textit{AuthorRes}}(y_i, y_j)$: A binary feature to reward the replying from p_i to p_j, if p_j's author a_j has replied to p_i's author a_i before. As the nature of threaded discussion, people discuss with each other in turn and make the conversation evolving. To track this reply/response pattern, we check when p_i replies to the author of p_j, whether post p_j has replied to the author of p_i before.

Author Preference $\textbf{\textit{AuthorPref}}(y_i, y_j)$: a binary feature to reward the pattern that post p_i and p_j reply to the same author when p_i and p_j are from one same author. The intention of this feature is to capture the preference of replying pattern between friends against other unknown users, who have not replied to each other before.

Content Propagation $\textbf{\textit{ConProp}}(y_i, y_j)$: cosine similarity between post p_i and p_j's parent post p_{y_j}, if p_i is replying to p_j. As we mentioned before, post contents are usually short and highly context dependent. In some cases, people have to borrow some content from one post's parent or even grandparents to understand its topic. To simulate this process, we assume that when p_i is replying to p_j, its content should be somehow similar to p_j's parent's content; in other words, we propagate information from p_j's parent to identify the possibility of p_i is replying to p_j.

Parent Propagation $\textbf{\textit{ParentProp}}(y_i, y_j)$: cosine similarity between post p_i and p_j, if they are replying to the same post. Similarity is a symmetric measurement, while replying relation is asymmetric. When two posts are similar to each other, they might be talking about similar topics in a replying relationship, or replying to the same post discussing the parallel aspects. The former is encoded in feature $\textbf{\textit{Sim}}(y_i)$, and $\textbf{\textit{ParentProp}}(y_i, y_j)$ is designed to catch the latter possibility.

Conversation Recency $\textbf{\textit{ConRecency}}(y_i, y_j)$: similar to $\textbf{\textit{Recency}}(y_i)$ by assuming a later post has a higher chance to be replied to. But in this feature, we calculate the normalized time gap between three consecutive posts in a conversation: if p_i replies to p_j while p_j replies to p_k, the time gap between p_j and p_k normalized by the time gap between p_i and p_k is used to measure the recency between y_i and y_j in the conversation. The intuition is that $\textbf{\textit{Recency}}(y_i)$ only considers the time gap in the whole thread (normalized by parent p_i's posting time to the root), while $\textbf{\textit{ConRecency}}(y_i, y_j)$ is designed to capture the recency in each possible subthread discussion. In other words, a newer post tends to join a recently hot discussion in the whole thread.

Table 1: Features for thread reconstruction task

Type	Feature
Node	$\textbf{\textit{Sim}}(y_i = j) = \cos(c_i, c_j)$
	$\textbf{\textit{First}}(y_i) = \begin{cases} 1 & y_i = 0 \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{Last}}(y_i) = \begin{cases} 1 & y_i = i-1 \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{AuthorRef}}(y_i = j) = \begin{cases} 1 & c_j \text{ contains } a_i\text{'s name} \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{Recency}}(y_i = j) = (t_j - t_0)/(t_i - t_0)$
	$\textbf{\textit{Self}}(y_i = j) = \begin{cases} -1 & a_i = a_j \\ 0 & \text{otherwise} \end{cases}$
Edge	$\textbf{\textit{Repeat}}(y_i, y_j) = \begin{cases} -1 & a_i = a_j, y_i = y_j \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{Jump}}(y_i, y_j) = \begin{cases} -1 & t_{y_i} < t_j, a_{y_i} = a_j, a_{y_j} = a_i \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{AuthorRes}}(y_i, y_j) = \begin{cases} 1 & a_{y_j} = a_i, y_i = j \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{AuthorPref}}(y_i, y_j) = \begin{cases} 1 & a_i = a_j, a_{p_i} = a_{p_j} \\ 0 & \text{otherwise} \end{cases}$
	$\textbf{\textit{ConProp}}(y_i, y_j) = \delta(y_i = j)\cos(c_i, c_{p_j})$
	$\textbf{\textit{ParentProp}}(y_i, y_j) = \delta(y_i = y_j)\cos(c_i, c_j)$
	$\textbf{\textit{ConRecency}}(y_i, y_j) = (t_j - t_{y_j})/(t_i - t_{y_j})$

Table 1 summarizes the definitions of these 13 different features, and Figure 2 gives an intuitive illustration of the effects introduced by those features.

The proposed edge features introduce many large-size cliques (in some extreme case the whole thread forms a fully connected graph), which make threadCRF more complex than the commonly used linear chain CRFs[10], skip-chain [16] and even 2D [22] CRFs. Exact inference is intractable in this situation, and we appeal to approximation methods to perform the Maximum a Posteriori inference and model learning.

4.3 Inference and Learning Method

We perform approximate Maximum a Posteriori (MAP) inference for Eq(2) based on a schedule for loopy belief propagation named Tree Reparameterizaton (TRP) [17], which breaks the loops in the induced graph into its spanning trees, and perform exact inference on such generated trees. TRP typically provides quick convergence and more accurate approximation results.

To learn the relative importance of the proposed features, we estimate the weights for each feature in a supervised manner. Given a training set of threads $T = \{X_1, X_2, \ldots, X_N\}$ with ground-truth replying relationship $R = \{\underline{Y_1}, \underline{Y_2}, \ldots, \underline{Y_N}\}$, we need to estimate the optimal model setting $\overline{\Theta} = \{\overline{\lambda_k}\}_{k=1}^{K}$, which maximizes the conditional likelihood defined in Eq(1) over the training set.

The log-likelihood function could be represented as:

$$L_\Theta = \sum_{n=1}^{N} \log p(Y_n|X_n, \Theta) \qquad (3)$$
$$= \sum_i \left[\Theta^T F(Y_n, X_n) - \log Z_\Theta(X_i)\right]$$

where $F(Y_n, X_n)$ are the accumulated feature values defined in Table 1, and $Z_\Theta(X_n) = \sum_Y \exp(\Theta^T F(Y, X_n))$.

To avoid overfitting, we penalize the likelihood with a spherical Gaussian weight prior $\lambda \sim N(0, \sigma^2)$. By taking the derivative of this object function, we get:

$$\nabla L_\Theta = \sum_{n=1}^{N} \left[F(Y_n, X_n) - E_{p_\Theta(Y|X_n)} F(Y, X_n)\right] - \frac{\lambda}{\sigma^2} \qquad (4)$$

The first part in Eq(4) is the accumulated empirical feature values in the training set, the second part is the expectation of the features' occurrences in the given training data, which could be efficiently calculated by TRP inferencer, and the last part is the regularization term from the Gaussian prior. This derivative is easy to understand: the maximum likelihood of the training data is reached when the empirical average of the global feature vector equals to its model expectation. L-BFGS algorithm is employed to optimize the object function Eq(3) by the gradient.

5. EXPERIMENTS

5.1 Forum Data Set

We collect 51,716 threads from three different online discussion forums with ground-truth replying relations: Apple Discussion (`http://discussions.apple.com`), Google Earth Community (`http://bbs.keyhole.com`) and CNET (`http://forums.cnet.com`) [5], for evaluation. Apple Discussion and CNET are two computer support forums, where people seek helps for the technical problems they encountered in hardware and software; while Google Earth Community is an entertainment focused forum, where people share interesting findings in the Google Earth software. Apple Discussion and Google Earth threads collections are available at `http://timan.cs.uiuc.edu/downloads.html`.

We perform simple preprocessing on these three collections to refine the evaluation corpus: 1) remove the threads with less than 3 posts, because we don't need to reconstruct the structure for such threads; 2) remove the threads with inconsistent posting time, i.e., earlier post replies to a later one; 3) remove a standard list of stop words [1] and punctuation from each post's content attribute. The basic statistics of the evaluation corpus are shown in Table 2.

Table 2: Summary of evaluation corpus

	#Thread	#User	Max #Posts	Posts /Thread	Tokens /Post
Apple	7,316	11,118	255	6.2	33.3
Google	8,636	6,936	251	7.4	17.6
CNET	15,886	14,382	38	4.9	46.1

From Table 2, we can get a brief sense of the differences between these three collections: 1) in Apple Discussion and CNET, people use more words to explain/answer the problems in their posts, while the posts in Google Earth are much shorter, where people only post links to the attractions they found in Google Earth with very few word descriptions; 2) users discuss more actively in Google Earth (1.25 thread/user) than other forums (0.66 thread/user in Apple Discussion and 1.10 thread/user in CNET).

5.2 Evaluation Criterion

Previous studies on replying relationship reconstruction only employ accuracy on the predicted edges (Acc$_{edge}$) as the evaluation criterion [11, 19]. However, we argue that such measurement is not sufficient to judge the goodness of the predicted structure: it only evaluates the point-wise predictions on each node, but loses sight on the predicted structure as a whole. We can use the example thread shown in Figure 1 to illustrate the defect in edge accuracy measurement.

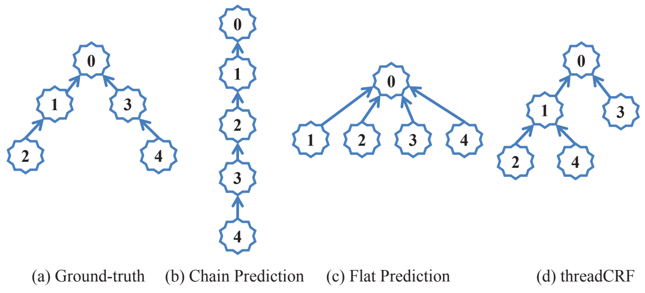


Figure 3: Example reconstruction results for thread shown in Figure 1

Figure 3 lists three different reconstruction results with the ground-truth structure. Both (b) and (d) achieved the same Acc$_{edge}$ of 0.75 (3 out of 4 predictions are correct), which is better than (c)'s Acc$_{edge}$ of 0.5. However, from the point of structural preservation, result (b) (a chain) is quite different from the original structure (a tree), and result (d) is arguably better from this perspective. Besides, when we take the actual discussion content into consideration, structure (b) misleads us: *Frank Miller2* in post 3 is actually replying to *dessto* in post 0 to give a possible solution in his/her first question, rather than to explain *dessto*'s further concern in post 2. Result (b) mistakenly connects post 3 to post 2, which changes the whole context. Therefore, to correctly maintain the discussion context in the scenario of conversational replying reconstruction, a mistake at higher level of the replying tree should incur more serious penalty than the one at a lower level.

To address this problem and more appropriately evaluate the quality of the predicted structure with respect to the ground-truth, we define a set of novel metrics. Formally, suppose $\overline{Y} = \{\overline{y}_0, \ldots, \overline{y}_{m-1}\}$ is the ground-truth structure for thread X, and $Y' = \{y'_0, \ldots, y'_{m-1}\}$ is the prediction. To evaluate how well the discussion context is preserved for each node in the tree, we define path accuracy Acc$_{path}$ as the proportion of correct path from each node to root:

$$\text{Acc}_{path} = \frac{\sum_{i=1}^{m-1} \delta\left[path_{\overline{Y}}(i) = path_{Y'}(i)\right]}{m-1} \qquad (5)$$

where $path_{\overline{Y}}(i)$ and $path_{Y'}(i)$ are the set of nodes lying in the path from node i to the root node in ground truth $\overline{Y}$

and prediction Y' respectively. For example, $path_{Y'}(4)$ in Figure 3.(b) is {4,3,2,1,0}.

Acc$_{path}$ measures if we can read from root to a particular post without missing a post, nor meeting an irrelevant one from other branch. To relax the strict whole path matching requirement, we compute the overlap between the path from one node to the root in ground-truth versus in the predicted path, and define path precision P$_{path}$ and recall R$_{path}$ as:

$$\mathrm{P}_{path} = \frac{\sum_{i=1}^{m-1} ||path_{\overline{Y}}(i) \subset path_{Y'}(i)||}{m-1} \quad (6)$$

$$\mathrm{R}_{path} = \frac{\sum_{i=1}^{m-1} ||path_{Y'}(i) \subset path_{\overline{Y}(i)}||}{m-1} \quad (7)$$

where $||S||$ is the size of set S.

Above path-based metrics emphasize the correct prediction of the nodes with more decedents at higher level of a tree: since we are counting path from every node to the root, higher level nodes will be involved multiple times by its decedents, in that case a mistake at higher levels would incur a more serious penalty.

Another important aspect that should be evaluated is how well a node's local structure is preserved. When one node is a branching node, it's crucial to recover all its child nodes to get the correct track of its sub-trees. For the same sake of allowing inexact match and distinguishing the missing case versus the incorrect-inserting case, we define node precision P$_{node}$ and recall R$_{node}$ as the overlap between the child node set of each node in ground-truth and prediction:

$$\mathrm{P}_{node} = \frac{\sum_{i=0}^{m-2} ||child_{\overline{Y}}(i) \subset child_{Y'}(i)||}{m-1} \quad (8)$$

$$\mathrm{R}_{node} = \frac{\sum_{i=0}^{m-2} ||child_{Y'}(i) \subset child_{\overline{Y}}(i)||}{m-1} \quad (9)$$

where $child_{\overline{Y}}(i)$ and $child_{Y'}(i)$ are node i's child node set in the corresponding structures. The summation is taken from 0 to m-2 since the last post does not have a child node.

As a commonly used compromise between precision and recall, we also define F-value for the proposed P$_{path}$, R$_{path}$ and P$_{node}$, R$_{node}$ as the harmonic mean of these two metrics. Now, we can take the example in Figure 3 to justify the proposed metrics: result (b) achieves Acc$_{path}$ of 0.5, F1$_{path}$ of 0.66 and F1$_{node}$ of 0.75, while result (d) receives Acc$_{path}$ of 0.75, F1$_{path}$ of 0.75 and F1$_{node}$ of 0.75, which is much closer to our intuitive expectation.

When we are evaluating in the corpus level, we could average all the above metrics in the thread level, known as Micro average, or in the whole corpus level, known as Macro average. In the following experiments, we employ both Micro/Marco performance as the evaluation metrics.

5.3 Reply Relation Reconstruction

To compare the effectiveness of the proposed method, we employ several baseline methods: 1) reply to Root Post (FIRST); 2) reply to Previous Post (LAST); 3) ranking by similarity (SIM), in which we use the same cosine similarity measure as ***sim(y$_i$)*** feature in threadCRF to select the precede most similar post as the parent post. Beside the unsupervised methods, we also use Seo et al.'s supervised Ranking SVM [13] as the baseline, where we use all our node features to train the Ranking SVM model, since Ranking SVM cannot deal with edge features. We use the default parameter setting as in Ranking SVM [7].

Because Apple Discussion and Google Earth are two types of online discussion sites, it would be interesting to compare the methods' performance on these two different domains. We use 75% threads in each collection as the training set and the rest threads for the testing purpose. The reported performance is calculated based on the evaluation criteria proposed in Section 5.2.

Table 3: Prediction performance on Apple

		FIRST	LAST	SIM	Ranking SVM	thread CRF
Acc$_{edge}$	Micro	0.4241	0.6598	0.5011	0.6977	**0.7350**
	Macro	0.5433	0.6912	0.5999	0.7534	**0.7858**
Acc$_{path}$	Micro	0.4241	0.4312	0.3918	0.4843	**0.5691**
	Macro	0.5433	0.5632	0.5322	0.6365	**0.6966**
P$_{path}$	Micro	**1.0000**	0.4312	0.6834	0.6681	0.7254
	Macro	**1.0000**	0.5632	0.7852	0.7780	0.8009
R$_{path}$	Micro	0.4241	**1.0000**	0.5524	0.7264	0.7559
	Macro	0.5433	**1.0000**	0.6744	0.8176	0.8564
F1$_{path}$	Micro	0.5956	0.6026	0.6110	0.6960	**0.7403**
	Macro	0.7041	0.7206	0.7256	0.7973	**0.8278**
P$_{node}$	Micro	**0.9097**	0.6598	0.7103	0.7675	0.8228
	Macro	**0.8909**	0.6912	0.7531	0.8090	0.8372
R$_{node}$	Micro	0.4846	**0.8461**	0.6339	0.8189	0.8315
	Macro	0.5774	0.8571	0.6974	0.8438	**0.8661**
F1$_{node}$	Micro	0.6324	0.7414	0.6699	0.7923	**0.8271**
	Macro	0.7006	0.7653	0.7242	0.8260	**0.8514**

Table 4: Prediction performance on Google

		FIRST	LAST	SIM	Ranking SVM	threadCRF
Acc$_{edge}$	Micro	0.4897	0.5524	0.5117	0.6285	**0.6308**
	Macro	0.5787	0.6325	0.6122	0.6920	**0.7147**
Acc$_{path}$	Micro	0.4897	0.3468	0.4016	0.4820	**0.5213**
	Macro	0.5787	0.5178	0.5514	0.6114	**0.6495**
P$_{path}$	Micro	**1.0000**	0.3468	0.5968	0.7000	0.7679
	Macro	**1.0000**	0.5178	0.7380	0.8157	0.8331
R$_{path}$	Micro	0.4897	**1.0000**	0.6593	0.6964	0.6745
	Macro	0.5787	**1.0000**	0.7480	0.7530	0.7727
F1$_{path}$	Micro	0.6575	0.5149	0.6265	0.6982	**0.7182**
	Macro	0.7332	0.6823	0.7430	0.7831	**0.8017**
P$_{node}$	Micro	**0.9231**	0.5524	0.7350	0.7468	0.8013
	Macro	**0.8941**	0.6325	0.7716	0.8001	0.8218
R$_{node}$	Micro	0.5510	**0.8168**	0.7029	0.7764	0.7523
	Macro	0.6122	**0.8381**	0.7422	0.7849	0.7995
F1$_{node}$	Micro	0.6901	0.6591	0.7186	0.7613	**0.7760**
	Macro	0.7267	0.7210	0.7566	0.7924	**0.8105**

From the results in Table 3 and Table 4, we can find that the proposed threadCRF model outperforms both the unsupervised and supervised baseline methods in these two different collections. The reason heuristic rule-based method FIRST has perfect P$_{path}$ is that all the posts are directly connected to the root post and no other posts can lie in between. The similar reasoning works for LAST's R$_{path}$ performance. However, these two simple heuristics could not capture the real replying structure in the data; as a result, their F1 performance is worse than the supervised learning methods. Besides, we can also find that similarity alone (**SIM**) is far from enough to predict the correct replying structures. The proposed threadCRF works better than Ranking SVM under most of performance metrics: threadCRF improves over 17.65% in Micro Acc$_{path}$ and 9.44% in Marco Acc$_{path}$ against Ranking SVM, which two are the strictest criteria to judge the goodness of the predicted structures. In addition, we perform the paired randomization test with $p<0.05$, which shows the statistical significance of improvements over the Ranking SVM. The comparison results with both the unsupervised heuristic rules and supervised Ranking SVM confirm the benefits of modeling the dependency among the posts in one thread, and the proposed threadCRF model correctly exploits the interactions in both posts and users by the introduced edge features.

When the thread size gets larger, i.e., more than 10 posts, it becomes harder for human to digest and understand the discussion content, and therefore, it is more necessary to perform automatically replying relation reconstruction in this situation. From another perspective, the evaluation performance might be biased by too many short threads, the comparison over long threads will better distinguish different methods' capability. In this experiment, we compare threadCRF with Ranking SVM on the long threads (more than 10 posts) to further investigate the effectiveness of the proposed method. We apply the models trained in previous experiment to analyze their performance only on the testing threads with more than 10 posts. Micro average is used over the selected performance metrics.

Table 5: Prediction performance on long threads

Data Set	Method	Acc_{edge}	Acc_{path}	F1_{path}	F1_{node}
Apple	RankingSVM	0.6207	0.4523	0.2314	0.7670
	threadCRF	**0.6513***	**0.5686***	**0.3338***	**0.7982***
Google	RankingSVM	**0.5414***	0.2938	0.5521	0.7171
	threadCRF	0.5076	**0.3634***	**0.6025***	**0.7366**

* indicates the improvement is significant ($p<0.05$).

In this testing set, we have 167 threads from Apple Discussion and 226 from Google Earth. Both methods' path-based performance drops compared with the results in Table 3 and Table 4, which is understandable, because the structures are getting more complex. Compared with Ranking SVM's performance, we can discover that threadCRF's performance improves more than that in all the testing cases: 25.71% and 23.67% in terms of Micro Acc_{path} in Apple Discussion and Google Earth accordingly.

5.4 Adaptability Evaluation

Although the proposed threadCRF works in a supervised manner, we want to test its adaptability when we are lack of such labeled training data. There are two scenarios in real situation: 1) we have a small set of labeled thread set from the target forum; 2) we do not any labeled threads from the target forum, but some labeled threads from other forums. In this experiment, we want to investigate our method's capability when applied in both of these two situations.

First, to test the method's dependency on the training data, we variate the training sets during the training phase. We select both Apple Discussion and Google Earth as the evaluation collection and use the same train/test separation as in Section 5.3. The volume of training set is gradually increased from empty to all the training threads. We select Ranking SVM model as the baseline method and compare their Micro Acc_{path} and Micro $F1_{node}$ performance in this experiment. When we have no training samples, we manually set all the features' weights to be 1 for both threadCRF and Ranking SVM.

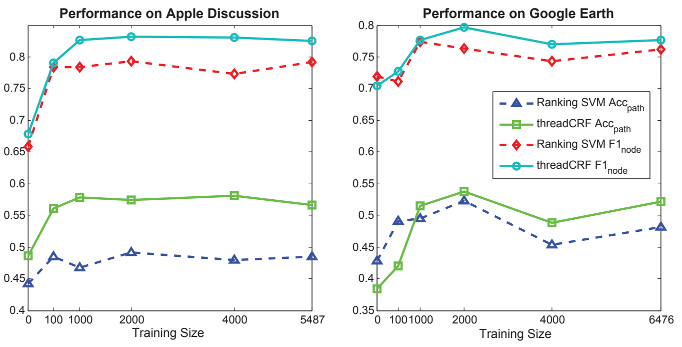


Figure 4: Performance on different training size

From Figure 4, we find with a small training set, 100 labeled threads, threadCRF achieves encouraging performance improvement against the default weight setting. As more training data is available, the performance converges for both methods. Another phenomenon we observe is that threadCRF's performance vibrates over different training sets. The reason is two folds: 1) the employed TRP inferencer in threadCRF is an approximation inference method, the estimated posterior probability varies as starting from different initial points; 2) the variances within the training set introduce noise into the learning phase, similar tendency could also be observed in Ranking SVM's performance.

Second, we perform a cross domain train/test to validate the learning method's generality. In this setting, we evaluate on all the three collections: we randomly select 2,000 threads for training and 2,000 threads for testing from the three data sets accordingly to make the comparison comparable. Besides evaluating the performance on the testing set from the same collections, we also apply the learned model on the other two testing sets from the different domains. Micro Acc_{path} is employed as the evaluation metric and we compare threadCRF with Ranking SVM in this experiment. In Table 6 and Table 7, columns indicate the testing set and rows indicate the training set.

Table 6: Cross domain evaluation on Ranking SVM by Acc_{path}

Train \ Test	Apple	Google	CNET
Apple	0.4730	0.4103	0.5553
Google	0.4757	0.4693	0.4896
CNET	0.4516	0.3724	0.6327

Table 7: Cross domain evaluation on threadCRFs by Acc_{path}

Train \ Test	Apple	Google	CNET
Apple	0.5527	0.4880	0.7172
Google	0.5270	0.4849	0.6919
CNET	0.5057	0.4313	0.6940

From the comparison results in Table 6 and Table 7, we find that threadCRF generalizes better than Ranking SVM in all the three data sets. Because Apple Discussion and

CNET are both computer technical forums, they share some properties in common. Therefore, the threadCRF model trained on Apple Discussion achieves promising performance on CNET, and vice versa. Instead, Google Earth is an entertainment focused forum, where the topics and users' interaction patterns are quite different from the other two forums. Ranking SVM trained on Google Earth doesn't perform as well as the threadCRF on the other two forums: 0.4880 of threadCRF v.s. 0.4103 of Ranking SVM on Apple Discussion and 0.4313 of threadCRF v.s. 0.3724 of Ranking SVM's on CNET. Besides, all the threadCRF's off-diagonal entries are better than Ranking SVM's, which indicates the proposed threadCRF possesses better generality capability than Ranking SVM.

It would be interesting to investigate the weights learned by threadCRF in different collections, from where we can discover interesting patterns in those domains. We normalize the weights from different domains by their corresponding largest weights to make them comparable, and illustrate part of the weights in Table 8.

Table 8: Normalized weights learned by threadCRF

	Sim(y_i)	***Self(y_i)***	***Repeat(y_i, y_j)***	***AuthorRes(y_i, y_j)***
Apple	1.0000	0.3102	0.1574	0.0527
Google	0.8441	0.3567	-0.0652	0.1343
CNET	0.5460	0.6218	0.6823	0.3145

From the learned feature weights, we can find ***Sim(y_i)*** is the most prominent feature in Apple Discussion. Because the topics in Apple Discussion are relatively more focused, mainly on Apple's products, replying post pairs tend to overlap more in their contents. In CNET, ***Self(y_i)*** (reply to oneself), ***Repeat(y_i, y_j)*** (repeatedly reply to the same post) and ***AuthorRes(y_i, y_j)*** (author response) are all relatively large. This indicates the conversational pattern in CNET is more significant than the other two forums. Another interesting observation is that the weight of ***Repeat(y_i, y_j)*** in Google is negative, which means users will reply to the same post they have replied to before. We manually checked the discussions in Google Earth Community and discovered that people repeatedly reply to the same post to provide more links and background of the findings in that post.

5.5 Applications

The replying relationship reconstructed by the proposed threadCRF model can be potentially useful for many further applications. Here we present two sample applications in forum search and community question answering to demonstrate the benefit of reconstructing the replying structure.

5.5.1 Forum Search

Seo et al. [13] demonstrate that using thread structures can lead to significant performance improvement in forum search compared to standard IR methods. In detail, they estimate a post language model by smoothing with the dialogue structure within the ground-truth replying structures.

Following the same line of retrieval model proposed in [13], we compare the retrieval performance on 1) post language model estimated on each individual post; 2) smoothing the post language model by the dialogue structure reconstructed by threadCRF and Ranking SVM; 3) smoothing by the whole thread, in order to justify whether the predicted structures could help the retrieval task.

In the CNET data set, Duan et al. manually selected 30 result documents for 30 different queries under the topic of "Computer & Internet", and annotated them as "relevant" or "irrelevant" [5]. 5-fold cross validation is employed to compare the retrieval performance, where the smoothing parameters in all the language models are exhaustively searched to maximize the MAP metric. MAP, P@1 and P@10 are employed as the evaluation criteria.

Table 9: Forum Search Performance on CNET

	MAP	P@1	P@10
Post Only	0.4893	0.4671	0.3532
Post + Thread	0.4959	0.4643	0.3634
RankingSVM	**0.5084**	0.4999	0.3964
threadCRF	0.5064	**0.5357**	**0.4036**

We can observe that the retrieval model estimated on the reconstructed thread structure generally improves the ranking performance over non-structure smoothing. The reason is that if we directly use all the posts in the thread to smooth the language model without carefully analyzing the semantic relationship between the posts, noise might be introduced into the ranking model and thus affect the retrieval performance, i.e., a degenerated P@1 for Post+Thread.

5.5.2 Community Question Answering

Online forum discussion is a valuable repository for question answering mining tasks. Previous studies on question answering mainly depend on content analysis techniques to find the most probable question-answer pairs [4, 3].

To confirm that the constructed conversational structure can help to detect the answers in the threaded discussion, we choose Apple Discussion as the evaluation collection, in which answer posts are explicitly labeled as "Solved" or "Helpful" by the users who raised the question in the discussion. We use the same train/test split as in Section 5.3.

The first step in question-answer pair extraction task is to identify the questions from the threaded discussion. There are many available techniques to perform such task [3, 6]. To get a clear sense of how the reconstructed structure can help us retrieve the answer posts in a threaded discussion, in this experiment, we assume the question posts are already identified before hand. To achieve it, we filter out the threads without the "Solved" or "Helpful" annotation in the testing set and assume the first post in the left 246 threads are the question posts.

In those question-answering oriented threads, we propose the following criterion to rank all the rest posts in the same thread by:

$$s(p_i) = \begin{cases} \text{child}(p_i) & \text{if root post's author has replied to } p_i \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

Intuitively, we assume a good answer should be the post which the question raiser has replied to, e.g., thanks for the replier or asks for further details, and receives many comments from others.

From Table 10, we can find that the naive content-based method, i.e., rank by similarity, doesn't work in this threaded discussion question answering environment, while replying structure based method achieves promising performance. Be-

Table 10: Performance on answer ranking

	similarity	Ranking SVM	threadCRF
MAP	0.2332	0.3969	**0.4402***
P@1	0.1799	0.6737	**0.6805***

* indicates the improvement is significant ($p<0.05$).

cause most posts in the threads are short, even though they are answering the questions in the previous posts, the content between question post and answer post overlaps little. The structure-based extraction method captures a strong indication for a good answer post from the perspective of user interactions, so that the precision is relatively high.

6. DISCUSSION

In this paper, we proposed a probabilistic graphical model based method, threadCRF, to solve the problem of replying relationship reconstruction in the online threaded discussion forums. By introducing various kind of features, the proposed threadCRF well captures both the long-range and short-range dependency among the posts and better predicts the replying relationship. To judge the quality of the predicted structures, we design a set of novel evaluation metrics. Experiment results on three different forum collections confirm the effectiveness of the proposed method, and the improvement over the baseline methods demonstrates the benefit of modeling the user interaction and long-range dependency in the threaded discussions.

Because the features employed in our method are not strictly limited to forum discussions, and they could be easily adopted to other domains, e.g., replying relationship mining in micro-weblogs, such as Twitter, and user wall posts and comments in social websites, such as Facebook. In addition, other advanced content analysis techniques, e.g., entity reconization/resolution and syntax parsing structure matching, can also be incorporated into our model as features to further enhance the performance.

7. ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-0713581, IIS-0905215, CNS 1028381, and by an AFOSR MURI Grant FA9550-08-1-0265, and the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA). We thank Huizhong Duan for providing the CNet data set and retrieval codes.

8. REFERENCES

[1] Onix text retrieval toolkit stopword list. `http://www.lextek.com/manuals/onix/stopwords1.html`.

[2] S. Bhatia and P. Mitra. Adopting Inference Networks for Online Thread Retrieval. In *Proceedings of the 24th AAAI*, pages 1300–1305, 2010.

[3] G. Cong, L. Wang, C. Lin, Y. Song, and Y. Sun. Finding question-answer pairs from online forums. In *Proceedings of the 31st SIGIR*, pages 467–474, 2008.

[4] S. Ding, G. Cong, C. Lin, and X. Zhu. Using conditional random fields to extract contexts and answers of questions from online forums. *Proceedings of ACL-08: HLT*, pages 710–718, 2008.

[5] H. Duan and C. Zhai. Exploiting Thread Structure to Improve Smoothing of Language Models for Forum Post Retrieval. In *Proceedings of the 33rd ECIR*, 2011.

[6] L. Hong and B. Davison. A classification-based approach to question answering in discussion boards. In *Proceedings of the 32nd SIGIR*, pages 171–178, 2009.

[7] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 8th KDD*, pages 133–142, 2002.

[8] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the 16th CIKM*, pages 919–922, 2007.

[9] D. Koller and N. Friedman. *Probabilistic graphical models.* MIT Press, 2009.

[10] J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML-2001*, pages 282–289, 2001.

[11] C. Lin, J. Yang, R. Cai, X. Wang, and W. Wang. Simultaneously modeling semantics and structure of threaded discussions: a sparse coding approach and its applications. In *Proceedings of the 32nd SIGIR*, pages 131–138, 2009.

[12] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing and management*, 24(5):513–523, 1988.

[13] J. Seo, W. Croft, and D. Smith. Online community search using thread structure. In *Proceedings of the 18th CIKM*, pages 1907–1910, 2009.

[14] D. Shen, Q. Yang, J. Sun, and Z. Chen. Thread detection in dynamic text message streams. In *Proceedings of 29th SIGIR*, pages 35–42, 2006.

[15] X. Shi, J. Zhu, R. Cai, and L. Zhang. User grouping behavior in online forums. In *Proceedings of the 15th KDD*, pages 777–786, 2009.

[16] C. Sutton and A. McCallum. *Collective segmentation and labeling of distant entities in information extraction.* 2004.

[17] M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on trees: message-passing and linear programming. *Information Theory, IEEE Transactions on*, 51(11):3697–3717, 2005.

[18] Y. Wang, M. Joshi, W. Cohen, and C. Rosé. Recovering implicit thread structure in newsgroup style conversations. In *ICWSM II*, 2008.

[19] Y. Wang and C. Rosé. Making conversational structure explicit: identification of initiation-response pairs within online discussions. In *Proceedings of HLT-NAACL 2010*, pages 673–676, 2010.

[20] G. Xu and W. Ma. Building implicit links from content for forum search. In *Proceedings of the 29th SIGIR*, pages 300–307, 2006.

[21] J. Zhang, M. Ackerman, and L. Adamic. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th WWW*, pages 221–230, 2007.

[22] J. Zhu, Z. Nie, J. Wen, B. Zhang, and W. Ma. 2d conditional random fields for web information extraction. In *Proceedings of the 22nd ICML*, pages 1044–1051, 2005.